

```
entry:
  %stktop_4 = alloca i8, i32 40, align 1
  %tos = ptrtoint ptr %stktop_4 to i64
  %0 = add i64 %tos, 12
  %RBP_N.28 = inttoptr i64 %0 to ptr
  %1 = add i64 %tos, 16
  %RBP_N.24 = inttoptr i64 %1 to ptr
  %2 = add i64 %tos, 24
  %RBP_N.16 = inttoptr i64 %2 to ptr
  %3 = add i64 %tos, 32
  %RBP_N.8 = inttoptr i64 %3 to ptr
  %4 = add i64 %tos, 0
  %RSP_P.0 = inttoptr i64 %4 to ptr
  store i64 3735928559, ptr %RSP_P.0, align 8
  %RBP = ptrtoint ptr %RSP_P.0 to i64
  store i64 %arg1, ptr %RBP_N.8, align 1
  store i64 %arg2, ptr %RBP_N.16, align 1
  store i64 %arg3, ptr %RBP_N.24, align 1
  store i32 %arg4, ptr %RBP_N.28, align 1
  store i32 0, ptr %stktop_4, align 1
  br label %bb.1
```

```
bb.1:
  %memload = load i32, ptr %stktop_4, align 1
  %5 = load i32, ptr %RBP_N.28, align 1
  %6 = sub i32 %memload, %5
  %7 = call { i32, i1 } @llvm.usub.with.overflow.i32(i32 %memload, i32 %5)
  %CF = extractvalue { i32, i1 } %7, 1
  %ZF = icmp eq i32 %6, 0
  %highbit = and i32 -2147483648, %6
  %SF = icmp ne i32 %highbit, 0
  %8 = call { i32, i1 } @llvm.ssub.with.overflow.i32(i32 %memload, i32 %5)
  %OF = extractvalue { i32, i1 } %8, 1
  %9 = and i32 %6, 255
  %10 = call i32 @llvm.ctpop.i32(i32 %9)
  %11 = and i32 %10, 1
  %PF = icmp eq i32 %11, 0
  %CmpSFOF_JGE = icmp eq i1 %SF, %OF
  br i1 %CmpSFOF_JGE, label %bb.3, label %bb.2
```

T	F
---	---

```
bb.3:
  ret i32 %memload
```

```
bb.2:
  %memload1 = load i64, ptr %RBP_N.16, align 1
  %memload2 = load i64, ptr %stktop_4, align 1
  %12 = trunc i64 %memload2 to i32
  %RCX = sext i32 %12 to i64
  %memref-basereg = add i64 %memload1, %RCX
  %13 = inttoptr i64 %memref-basereg to ptr
  %memload3 = load i32, ptr %13, align 1
  %14 = trunc i32 %memload3 to i8
  %EDX = sext i8 %14 to i32
  %memload4 = load i64, ptr %RBP_N.24, align 1
  %memload5 = load i64, ptr %stktop_4, align 1
  %15 = trunc i64 %memload5 to i32
  %RCX6 = sext i32 %15 to i64
  %memref-basereg7 = add i64 %memload4, %RCX6
  %16 = inttoptr i64 %memref-basereg7 to ptr
  %memload8 = load i32, ptr %16, align 1
  %17 = trunc i32 %memload8 to i8
  %ESI = sext i8 %17 to i32
  %EDX9 = mul nsw i32 %EDX, %ESI
  %memload10 = load i64, ptr %RBP_N.8, align 1
  %memload11 = load i64, ptr %stktop_4, align 1
  %18 = trunc i64 %memload11 to i32
  %RCX12 = sext i32 %18 to i64
  %memref-basereg13 = add i64 %memload10, %RCX12
  %19 = trunc i32 %EDX9 to i8
  %20 = inttoptr i64 %memref-basereg13 to ptr
  store i8 %19, ptr %20, align 1
  %memload14 = load i32, ptr %stktop_4, align 1
  %EAX = add i32 %memload14, 1
  %21 = call { i32, i1 } @llvm.uadd.with.overflow.i32(i32 %memload14, i32 1)
  %CF15 = extractvalue { i32, i1 } %21, 1
  %22 = and i32 %EAX, 255
  %23 = call i32 @llvm.ctpop.i32(i32 %22)
  %24 = and i32 %23, 1
  %PF16 = icmp eq i32 %24, 0
  %ZF17 = icmp eq i32 %EAX, 0
  %highbit18 = and i32 -2147483648, %EAX
  %SF19 = icmp ne i32 %highbit18, 0
  %25 = call { i32, i1 } @llvm.sadd.with.overflow.i32(i32 %memload14, i32 1)
  %OF20 = extractvalue { i32, i1 } %25, 1
  store i32 %EAX, ptr %stktop_4, align 1
  br label %bb.1
```

CFG for 'mix_comp' function